Object-Oriented Programming I (CSE 121)
Week # 02
(YouTube Lecture: 04)
C++ Mathematical Functions

Objective(s)

In this chapter, you will learn:

- A list of popular mathematical functions
- #include<cmath>
- #include<algorithm>

C++ has many functions that allows you to perform mathematical tasks on numbers. The C++ <cmath> header file declares a set of functions to perform mathematical operations. A list of popular Math functions is given below:

1. **pow()**: The pow() function computes a base number raised to the power of exponent number.
   [Mathematics] $base^{exponent}$= pow(base, exponent) [In C++ programming]
2. **sqrt()**: The sqrt(x) function can be used to find the square root of a number.
   [Mathematics] $\sqrt{x}$ = sqrt(x) [In C++ programming]
3. **cbrt()**: The cbrt(x) function can be used to find the cube root of a number.
   [Mathematics] $\sqrt[3]{x}$ = cbrt(x) [In C++ programming]
4. **floor()**: The floor() function rounds off the given value to the closest integer which is less than the given value.
5. **ceil()**: The ceil() function rounds off the given value to the closest integer which is more than the given value.
6. **round()**: The round() function rounds given number to the closest integer.
7. **trunc()**: The trunc() function rounds removes digits after decimal point.
8. **abs()**: The abs() function returns the absolute value of the argument.
   [Mathematics] $|x|$ = abs(x) [In C++ programming]
9. **max()**: The max(x,y) function can be used to find the highest value of x and y.
10. **min()**: The min(x,y) function can be used to find the lowest value of x and y.

1. The pow() function computes a base number raised to the power of exponent number.

pow.cpp

```cpp
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    double b,e,r;//b: base,e: exponent,r: result
    cout << "Enter the base value: ";
    cin >> b;
    cout << "Enter the exponent value: ";
    cin >> e;
        r = pow(b,e);
    cout << b << "^" << e << " = " << r << endl;
    return 0;
}
```

C:\Users\Bulbul\Desktop\math\pow.ex

```
Enter the base value: 2
Enter the exponent value: 10
2^10 = 1024
```

2. The sqrt(x) function can be used to find the square root of a number.

pow.cpp sqrt.cpp

```cpp
#include<iostream>
#include<iomanip>
#include<cmath>
using namespace std;

int main(){
    double n1 = 225.0;
    double n2 = 300.0;
    cout << fixed << setprecision(8);
    cout << sqrt(n1) << endl;
    cout << sqrt(n2) << endl;
    return 0;
}
```

C:\Users\Bulbul\D

```
15.00000000
17.32050808
```

3. The cbrt(x) function can be used to find the cube root of a number.

cbrt.cpp

```cpp
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    double n1 = 27, n2 = 17;
    cout << cbrt(n1) << endl;
    cout << cbrt(n2) << endl;

    cout << pow(n1,1.0/3.0) << endl;
    cout << pow(n2,1.0/3.0) << endl;
    return 0;
}
```

C:\User

```
3
2.57128
3
2.57128
```

4. The floor() function rounds off the given value to the closest integer which is less than the given value

floor.cpp

```cpp
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    double x = 1.311, y = 1.500, z = 1.822;
    cout << "x = " << floor(x) << endl;
    cout << "y = " << floor(y) << endl;
    cout << "z = " << floor(z) << endl;
    double a = -1.311, b = -1.500, c = -1.822;
    cout << "a = " << floor(a) << endl;
    cout << "b = " << floor(b) << endl;
    cout << "c = " << floor(c) << endl;
    return 0;
}
```

C:\Us

```
x = 1
y = 1
z = 1
a = -2
b = -2
c = -2
```

5. The ceil() function rounds off the given value to the closest integer which is more than the given value.

ceil.cpp

```cpp
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    double x = 1.311, y = 1.500, z = 1.822;
    cout << "x = " << ceil(x) << endl;
    cout << "y = " << ceil(y) << endl;
    cout << "z = " << ceil(z) << endl;
    double a = -1.311, b = -1.500, c = -1.822;
    cout << "a = " << ceil(a) << endl;
    cout << "b = " << ceil(b) << endl;
    cout << "c = " << ceil(c) << endl;
    return 0;
}
```

C:\Use

```
x = 2
y = 2
z = 2
a = -1
b = -1
c = -1
```

6. The round() function rounds given number to the closest integer.

round.cpp

```cpp
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    double x = 1.311, y = 1.500, z = 1.822;
    cout << "x = " << round(x) << endl;
    cout << "y = " << round(y) << endl;
    cout << "z = " << round(z) << endl;
    double a = -1.311, b = -1.500, c = -1.822;
    cout << "a = " << round(a) << endl;
    cout << "b = " << round(b) << endl;
    cout << "c = " << round(c) << endl;
    return 0;
}
```

C:\Us

```
x = 1
y = 2
z = 2
a = -1
b = -2
c = -2
```

7. The trunc() function rounds removes digits after decimal point.

trunc.cpp

```cpp
#include<iostream>
#include<cmath>
using namespace std;

int main(){
    double x = 1.311, y = 1.500, z = 1.822;
    cout << "x = " << trunc(x) << endl;
    cout << "y = " << trunc(y) << endl;
    cout << "z = " << trunc(z) << endl;
    double a = -1.311, b = -1.500, c = -1.822;
    cout << "a = " << trunc(a) << endl;
    cout << "b = " << trunc(b) << endl;
    cout << "c = " << trunc(c) << endl;
    return 0;
}
```

C:\Use

```
x = 1
y = 1
z = 1
a = -1
b = -1
c = -1
```

8. abs(): The abs() function returns the absolute value of the argument.

abs.cpp

C:\ProgramData\Microsoft\Windows\Start Menu\Programs\CodeBlocks\Start here

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main(){
    double x = -97.99;
    cout << "|" << x << "| = " << abs(x);
    cout << endl;
    int y = -101;
    cout << "|" << y << "| = " << abs(y);
    cout << endl;
    return 0;
}
```

C:\Users\Bulbul\De

```
|-97.99| = 97.99
|-101| = 101
```

9. max(): The max(*x*,*y*) function can be used to find the highest value of *x* and *y*.

max.cpp

```cpp
#include <iostream>
#include <algorithm>
using namespace std;

int main(){

    cout << "max(10,20) = " << max(10,20) << endl;
    cout << "max(20,10) = " << max(20,10) << endl;
    cout << "max('a','z') = " << max('a','z') << endl;
    cout << "max('k','a') = " << max('k','a') << endl;
    cout << "max(12.58,5.75) = " << max(12.58,5.75) << endl;
    return 0;
}
```

C:\Users\Bulbul\Desktop\math

```
max(10,20) = 20
max(20,10) = 20
max('a','z') = z
max('k','a') = k
max(12.58,5.75) = 12.58
```

10. min(): The min(*x*,*y*) function can be used to find the lowest value of *x* and *y*.

min.cpp

```cpp
#include <iostream>
#include <algorithm>
using namespace std;

int main(){

    cout << "min(10,20) = " << min(10,20) << endl;
    cout << "min(20,10) = " << min(20,10) << endl;
    cout << "min('a','z') = " << min('a','z') << endl;
    cout << "min('k','a') = " << min('k','a') << endl;
    cout << "min(12.58,5.75) = " << min(12.58,5.75) << endl;
    return 0;
}
```

C:\Users\Bulbul\Desktop\math

```
min(10,20) = 10
min(20,10) = 10
min('a','z') = a
min('k','a') = a
min(12.58,5.75) = 5.75
```

. . . .